L 12

```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
SSS              YYY         YYY  SSS
SSS              YYY         YYY  SSS
SSS              YYY         YYY  SSS
SSS                 YYY   YYY     SSS
SSS                 YYY   YYY     SSS
SSS                 YYY   YYY     SSS
   SSSSSSSSS           YYY           SSSSSSSSS
   SSSSSSSSS           YYY           SSSSSSSSS
   SSSSSSSSS           YYY           SSSSSSSSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
```

```
_$
Ps
--
YZ
Z$
Z$
Z$
Z$
Z$
Z$
Z$
Z$
Z$
Z$
Z$
Z$
```

```
**FILE**ID**DISMOUNT                                    D  3




DDDDDDDD      IIIIII      SSSSSSSS  MM      MM    OOOOOO    UU      UU  NN      NN  TTTTTTTTTT
DDDDDDDD      IIIIII      SSSSSSSS  MM      MM    OOOOOO    UU      UU  NN      NN  TTTTTTTTTT
DD      DD      II      SS          MMMM  MMMM  OO      OO  UU      UU  NN      NN      TT
DD      DD      II      SS          MMMM  MMMM  OO      OO  UU      UU  NN      NN      TT
DD      DD      II      SS          MM  MM  MM  OO      OO  UU      UU  NNNN    NN      TT
DD      DD      II      SS          MM  MM  MM  OO      OO  UU      UU  NNNN    NN      TT
DD      DD      II        SSSSSS    MM      MM  OO      OO  UU      UU  NN  NN  NN      TT
DD      DD      II        SSSSSS    MM      MM  OO      OO  UU      UU  NN  NN  NN      TT
DD      DD      II              SS  MM      MM  OO      OO  UU      UU  NN    NNNN      TT
DD      DD      II              SS  MM      MM  OO      OO  UU      UU  NN    NNNN      TT
DD      DD      II              SS  MM      MM  OO      OO  UU      UU  NN      NN      TT         ....
DD      DD      II              SS  MM      MM  OO      OO  UU      UU  NN      NN      TT         ....
DDDDDDDD      IIIIII    SSSSSSSS    MM      MM    OOOOOO    UUUUUUUUUU  NN      NN      TT         ....
DDDDDDDD      IIIIII    SSSSSSSS    MM      MM    OOOOOO    UUUUUUUUUU  NN      NN      TT         ....


LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

DI
VC

```
0000            .TITLE  DISMOUNT - DISMOUNT A MOUNTED MASS STORAGE VOLUME
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000
0000    ;++
0000    ;
0000    ; FACILITY:
0000    ;
0000    ;       MASS STORAGE DEVICE MANAGEMENT SUBROUTINES
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       THIS ROUTINE DISMOUNTS THE INDICATED DEVICE.
0000    ;
0000    ;
0000    ; ENVIRONMENT:
0000    ;
0000    ;       VAX/VMS EXEC
0000    ;       MODE = KERNEL
0000    ;
0000    ;
0000    ; AUTHOR:  ANDREW C. GOLDSTEIN, CREATION DATE:  2-NOV-1977  14:10
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-019 CDS0001         Christian D. Saether    28-Aug-1984
0000    ;               Ignore SS$_VALNOTVALID errors when converting device
0000    ;               lock.
0000    ;
0000    ;       V03-018 HH0049          Hai Huang               16-Aug-1984
0000    ;               Call IOC$DALLOC_DMT routine to deallocate the device
0000    ;               on dismount of a foreign volume.
0000    ;
0000    ;       V03-017 ACG0441         Andrew C. Goldstein,    13-Aug-1984  10:17
```

```
DI
Ps

PS
--
.
$A
Y$

Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
11
Th
56
42

Ma
--
_$
_$
TC

24

Th

MA
```

```
0000    ;               Issue both an IO$_UNLOAD and IO$_AVAILABLE to correctly
0000    ;               release tape drives.
0000    ;
0000    ;       V03-016 ACG0441         Andrew C. Goldstein,    8-Aug-1984  11:33
0000    ;               Rework foreign volume dismount; locate all code in
0000    ;               this module. General code cleanup.
0000    ;
0000    ;       V03-015 TMK0001         Todd M. Katz            21-Apr-1984
0000    ;               When deleting the logical name associated with a mounted volume,
0000    ;               delete the logical name block by calling LNM$DELETE_LNMB
0000    ;               instead of LNM$DELETE. Doing so will ensure that this deletion
0000    ;               takes place as if the system service $DELLNM had been called
0000    ;               to delete the logical name. In other words, not only will the
0000    ;               target logical name be deleted, but so will all outer access
0000    ;               mode aliases.
0000    ;
0000    ;       V03-014 LMP0221         L. Mark Pilant,         30-Mar-1984  13:48
0000    ;               Change UCB$L_OWNUIC to ORB$L_OWNER and UCB$W_VPROT to
0000    ;               ORB$W_PROT.
0000    ;
0000    ;       V03-013 ACG0371         Andrew C. Goldstein,    11-Nov-1983  9:32
0000    ;               Set PHY_IO in PCB privilege mask instead of PHD
0000    ;
0000    ;       V03-012 LY0427          Larry Yetto             5-OCT-1983 14:51:12
0000    ;               If the DELJNL service call to delete the RU journal fails
0000    ;               then deassign the journal channel
0000    ;
0000    ;       V01-011 TCM0005         Trudy C. Matthews       22-Sep-1983
0000    ;               If device is to be deallocated on dismount, don't do it here.
0000    ;               Wait until last channel deassign instead.  This keeps the
0000    ;               device allocated and the lock present until all activity
0000    ;               has ceased from this mount.
0000    ;
0000    ;       V03-010 TCM0004         Trudy C. Matthews       07-Sep-1983
0000    ;               Fix bug that caused foreign disks not to be unloaded on
0000    ;               dismount.
0000    ;
0000    ;       V03-009 TCM0003         Trudy C. Matthews       22-Aug-1983
0000    ;               Undo change made in TCM0001.  If a device is dismounted and
0000    ;               there are still channels assigned to it, we just want to
0000    ;               deallocate the local UCB.  The cluster-wide lock (if it
0000    ;               exists) will be dequeued when the last channel is de-assigned.
0000    ;
0000    ;       V03-008 TCM0002         Trudy C. Matthews       22-Jun-1983
0000    ;               Decrement refcount when a disk is dismounted.  MOUNT has
0000    ;               been changed to increment the refcount while the disk
0000    ;               is mounted.
0000    ;
0000    ;       V03-007 ADE9006         Alan D. Eldridge        01-MAy-1983
0000    ;               Restore PCB address (R4) on dismount of foreign devices.
0000    ;
0000    ;       V03-006 STJ3103         Steven T. Jeffreys,     27-Apr-1983
0000    ;               Delete RUJ on dismount.
0000    ;
0000    ;       V03-005 DMW4034         DMWalp                  26-May-1983
0000    ;               Intergate new logical name structures.
0000    ;
```

```
0000      ;          V03-004 TCM0001          Trudy C. Matthews         21-Apr-1982
0000      ;                  Call routine EXE$DALLOC_DEV to deallocate a device.  This
0000      ;                  routine handles cluster device deallocation correctly.
0000      ;
0000      ;          V03-003 PHL0101          Peter H. Lipman           20-Jun-1982
0000      ;                  $QIOW now synchronizes the EFN and IOSB parameters
0000      ;                  correctly.  Eliminate the synchronization code here.
0000      ;
0000      ;          V03-002 STJ0257          Steven T. Jeffreys,       12-Apr-1982
0000      ;                  - Do not mung device allocation access mode.
0000      ;                  - Make code AST reentrant.  This includes the addition
0000      ;                    of the local subroutine DO_IO.
0000      ;
0000      ;          V03-001 STJ0229          Steven T. Jeffreys,       23-Mar-1982
0000      ;                  Clear the 'mount verification possible' bit in the VCB
0000      ;                  so that $DISMOU will succeed even if no volume is present
0000      ;                  in the drive (as in version 2).
0000      ;
0000      ;          V02-008 ACG0248          Andrew C. Goldstein,      23-Dec-1981  11:56
0000      ;                  Fix logical name interlocks
0000      ;
0000      ;          V02-007 ACG0226          Andrew C. Goldstein,      24-Nov-1981  22:29
0000      ;                  Issue IO$_AVAILABLE on DISMOUNT/NOUNLOAD
0000      ;
0000      ;          V0006   STJ0138          Steven T. Jeffreys,   12-Nov-1981
0000      ;                  Use IOC$CVT_DEVNAM to format the device name.
0000      ;
0000      ;          V0005   ACG0062          Andrew C. Goldstein,  16-Oct-1979  13:53
0000      ;                  Unload volumes mounted foreign on dismount
0000      ;
0000      ;          V0004   ACG0003          Andrew C. Goldstein,  1-Feb-1979  11:07
0000      ;          Add handling of dummy MTL entry for volume set
0000      ;
0000      ;     Andrew C. Goldstein, 12-Jul-78  20:08
0000      ;     V0003 - ADD ERROR LOG ENTRY FOR FOREIGN DISMOUNT
0000      ;
0000      ;**
0000
0000      ;
0000      ; Define system control blocks
0000      ;
0000               $DDBDEF                           ; DDB
0000               $DEVDEF                           ; device characteristics bits
0000               $DCDEF                            ; define device types
0000               $EMBETDEF                         ; define error log message codes
0000               $EMBVMDEF                         ; define error log buffer format
0000               $IODEF                            ; define I/O function codes
0000               $IPLDEF                           ; define IPL definitions
0000               $LCKDEF                           ; define lock manager values
0000               $LKIDEF                           ; define codes for $GETLKI
0000               $ORBDEF                           ; object's rights block offsets
0000               $PCBDEF                           ; process control block
0000               $PRDEF                            ; processor register codes
0000               $PRVDEF                           ; privilege bit definitions
0000               $MTLDEF                           ; mounted volume list entry
0000               $SSDEF                            ; system service codes
0000               $UCBDEF                           ; UCB
```

```
                  0000            $VCBDEF                                 ; VCB
                  0000     ;
                  0000     ; Local storage allocated on stack (addressed off R3)
                  0000     ;
         00000020 0000     NAME_LENGTH       = 32                         ; length of device name buffer
         00000000 0000     CHANNEL           = 0                          ; channel number
         00000004 0000     DEVICE_NAME       = 4                          ; string descriptor of device name
         0000000C 0000     NAME_STRING       = 12                         ; device name string buffer
                  0000
         0000002C 0000     LOCAL_SIZE        = 44                         ; total size of stack locals
                  0000
              00000000            .PSECT  Y$DISMOUNT
```

```
                      0000  ;++
                      0000  ;
                      0000  ; FUNCTIONAL DESCRIPTION:
                      0000  ;
                      0000  ;       This routine dismounts the indicated mounted volume list entry.
                      0000  ;       The MTL and logical name, if it still exists, are deleted, and the
                      0000  ;       volume share count is decremented. If the share count goes to
                      0000  ;       zero, the volume itself is dismounted.
                      0000  ;
                      0000  ; CALLING SEQUENCE:
                      0000  ;       JSB IOC$DISMOUNT
                      0000  ;
                      0000  ; INPUT PARAMETERS:
                      0000  ;       R3 = LBC to unload volume
                      0000  ;            LBS to not unload
                      0000  ;       R4 = address of process PCB
                      0000  ;       R6 = address of mounted volume list entry
                      0000  ;
                      0000  ; IMPLICIT INPUTS:
                      0000  ;       IPL  -  IPL$_ASTDEL
                      0000  ;
                      0000  ; OUTPUT PARAMETERS:
                      0000  ;       R0-R2,R6 smashed, other registers preserved
                      0000  ;
                      0000  ; IMPLICIT OUTPUTS:
                      0000  ;       NONE
                      0000  ;
                      0000  ; ROUTINE VALUE:
                      0000  ;       SS$_NORMAL,SS$_NOIOCHAN
                      0000  ;
                      0000  ; SIDE EFFECTS:
                      0000  ;       Volume dismounted: logical name & MTL deallocated, VCB gone or soon
                      0000  ;       to go, ACP process may become deleted
                      0000  ;
                      0000  ;--
                      0000
                      0000
                      0000  IOC$DISMOUNT::
              38  BB  0000          PUSHR   #^M<R3,R4,R5>           ; save registers
        55 0C A6  D0  0002          MOVL    MTL$L_UCB(R6),R5        ; get UCB address
           10 A6  D5  0006          TSTL    MTL$L_LOGNAME(R6)       ; test address of logical name
              1F  13  0009          BEQL    10$                     ; branch if none
                      000B          DSBINT  S^#IPL$_ASTDEL
      00000000'EF 16  0011          JSB     LNM$LOCKW               ; lock the table
        51 10 A6  D0  0017          MOVL    MTL$L_LOGNAME(R6),R1    ; get address of logical name
      00000000'EF 16  001B          JSB     LNM$DELETE_LNMB         ; delete the logical name
      00000000'EF 16  0021          JSB     LNM$UNLOCK              ; and unlock the table
                      0027          ENBINT
                      002A
        7E 0B A6  9A  002A  10$:    MOVZBL  MTL$B_STATUS(R6),-(SP)  ; save MTL entry status byte
           50 56  D0  002E          MOVL    R6,R0                   ; get MTL address in R0
              05  18  0031          BGEQ    20$                     ; branch if process space address
           FFCA'  30  0033          BSBW    EXE$DEAPAGED            ; deallocate to system paged pool
              0E  11  0036          BRB     30$
        51 08 A6  3C  0038  20$:    MOVZWL  MTL$W_SIZE(R6),R1       ; get block size
   53 00000000'9F DE  003C          MOVAL   @#CTL$GQ_ALLOCREG,R3    ; and process allocation list head
           FFBA'  30  0043          BSBW    EXE$DEALLOCATE          ; and deallocate to process pool
```

```
                              0046
                              0046      ;
                              0046      ; Now lock the I/O database mutex and decrement the volume share count.
                              0046      ; If it goes to zero, mark the UCB for dismount.
                              0046      ;
                              0046                ASSUME  MTL$V_VOLSET EQ 0
            0F 8E      E8     0046      30$:      BLBS    (SP)+,40$                   ; branch if MTL entry was for volume set
             FFB4'     30     0049                BSBW    SCH$IOLOCKW                 ; lock I/O database
      50   34 A5       D0     004C                MOVL    UCB$L_VCB(R5),R0            ; and VCB address
           4C A0       B7     0050                DECW    VCB$W_MCOUNT(R0)            ; decrement mount count
              0C       13     0053                BEQL    50$                         ; branch if now idle
             FFA8'     30     0055                BSBW    SCH$IOUNLOCK                ; else unlock I/O database
                              0058      40$:      SETIPL  #0
      50      01       D0     005B                MOVL    #SS$_NORMAL,R0              ; set success
            0077       31     005E                BRW     130$                        ; and get out
                              0061
   00 38 A5   15       E2     0061      50$:      BBSS    #DEV$V_DMT,UCB$L_DEVCHAR(R5),60$ ; set mark for dismount
         05 6E         E9     0066      60$:      BLBC    (SP),70$                    ; branch if volume to be unloaded
   00 64 A5   0C       E5     0069                BBCC    #UCB$V_UNLOAD,UCB$W_STS(R5),70$ ; else clear unload bit
              04       8A     006E      70$:      BICB2   #<1@VCB$V_MOUNTVER>,-       ; clear MV bit in the VCB
         53 A0                0070                        VCB$B_STATUS2(R0)
   00 64 A5   09       E4     0072      80$:      BBSC    #UCB$V_MOUNTING,UCB$W_STS(R5),90$ ; clean up status bits
             FF86'     30     0077      90$:      BSBW    SCH$IOUNLOCK                ; unlock the I/O database
                              007A                SETIPL  #0
                              007D
                              007D      ;
                              007D      ; Assign a channel to the device. If it is mounted Files-11, issue a dismount
                              007D      ; QIO. (If it is mounted foreign, deassigning the channel will complete the
                              007D      ; cleanup).
                              007D      ;
        5E    2C       C2     007D                SUBL    #LOCAL_SIZE,SP              ; allocate local storage on stack
        53    5E       D0     0080                MOVL    SP,R3
        50    20       D0     0083                MOVL    #NAME_LENGTH,R0             ; set name buffer length
      51   0C A3       DE     0086                MOVAL   NAME_STRING(R3),R1          ; set name buffer address
   08 A3      51       D0     008A                MOVL    R1,DEVICE_NAME+4(R3)        ; copy address to descriptor
              54       D4     008E                CLRL    R4                          ; get node + device name
             FF6D'     30     0090                BSBW    IOC$CVT_DEVNAM              ; format the device name
   04 A3      51       D0     0093                MOVL    R1,DEVICE_NAME(R3)          ; save resultant string length
              63       D4     0097                CLRL    CHANNEL(R3)                 ; init channel number
                              0099                $ASSIGN_S CHAN=CHANNEL(R3),-        ; and assign a channel to the device
                              0099                        DEVNAM=DEVICE_NAME(R3)
           2B 50       E9     00A7                BLBC    R0,120$                     ; if this fails, we will have a hung device
                              00AA
              18       E1     00AA                BBC     #DEV$V_FOR,-                ; if BC then not foreign
        08 38 A5              00AC                        UCB$L_DEVCHAR(R5),100$
      54   30 AE       D0     00AF                MOVL    LOCAL_SIZE+4(SP),R4         ; recover PCB address
              26       10     00B3                BSBB    FOREIGN                     ; dismount foreign device
              14       11     00B5                BRB     110$                        ; continue
                              00B7
     00000438 8F       DD     00B7      100$:     PUSHL   #<IO$_ACPCONTROL!IO$M_DMOUNT>
              63       DD     00BD                PUSHL   CHANNEL(R3)                 ; push channel number
     0255'CF  02       FB     00BF                CALLS   #2,W^DO_IO                  ; issue the dismount QIO
      54   04 A3       7E     00C4                MOVAQ   DEVICE_NAME(R3),R4          ; get address of device name descriptor
            01B3       30     00C8                BSBW    DELETE_RUJ                  ; delete the recovery unit journal (ruj)
                              00CB      110$:     $DASSGN_S  CHAN=CHANNEL(R3)         ; deassign the channel
        5E    2C       C0     00D5      120$:     ADDL    #LOCAL_SIZE,SP              ; restore stack pointer
                              00D8                                                    ;
```

```
               38   BA  00D8    299 130$:    POPR    #^M<R3,R4,R5>             ; restore registers
                    05  00DA    300          RSB
```

```
                                00DB              .SBTTL  DISMOUNT FOREIGN DEVICE
                                00DB   ;
                                00DB   ; Foreign devices are dismounted on the spot, with no interlock checks.
                                00DB   ; The reason for this is that the only event we could defer the dismount
                                00DB   ; to is last channel deassign, which is not a suitable location.
                                00DB   ;
                                00DB   ; Construct and send the error log message signalling dismount.
                                00DB   ;
                                00DB
                                00DB              .ENABLE LSB
                                00DB   FOREIGN:                                           ; dismount foreign device
                51   3E    9A   00DB              MOVZBL  #EMB$K_VM_LENGTH,R1             ; length of error log message
                   FF1F'   30   00DE              BSBW    ERL$ALLOCEMB                    ; allocate an error log buffer
                   41 50   E9   00E1              BLBC    R0,10$                          ; branch if failure
                                00E4                                                      ;
                      3C   BB   00E4              PUSHR   #^M<R2,R3,R4,R5>                ; save address of buffer and R3, R4, R5
                                00E6              ASSUME  EMB$L_VM_ERRCNT EQ EMB$L_VM_OWNUIC+4
                                00E6              ASSUME  EMB$L_VM_OPRCNT EQ EMB$L_VM_ERRCNT+4
                                00E6              ASSUME  EMB$W_VM_UNIT   EQ EMB$L_VM_OPRCNT+4
                                00E6              ASSUME  EMB$B_VM_NAMLNG EQ EMB$W_VM_UNIT+2
                                00E6              ASSUME  EMB$T_VM_NAMTXT EQ EMB$B_VM_NAMLNG+1
                                00E6              ASSUME  EMB$W_VM_VOLNUM EQ EMB$T_VM_NAMTXT+15
                                00E6              ASSUME  EMB$W_VM_NUMSET EQ EMB$W_VM_VOLNUM+2
                                00E6              ASSUME  EMB$T_VM_LABEL  EQ EMB$W_VM_NUMSET+2
                                00E6
      04 A2    0041 8F     B0   00E6              MOVW    #EMB$K_VD,EMB$W_VM_ENTRY(R2) ; message code = dismount
               52   10     C0   00EC              ADDL    #EMB$L_VM_OWNUIC,R2             ; point to entries to be filled in
          50    1C A5      D0   00EF              MOVL    UCB$L_ORB(R5),R0                ; get ORB address
               82   60     D0   00F3              MOVL    ORB$L_OWNER(R0),(R2)+           ; volume owner UIC
          82    0082 C5    3C   00F6              MOVZWL  UCB$W_ERRCNT(R5),(R2)+          ; volume error count
             82   70 A5    D0   00FB              MOVL    UCB$L_OPCNT(R5),(R2)+           ; volume operation count
             82   54 A5    B0   00FF              MOVW    UCB$W_UNIT(R5),(R2)+            ; unit number
       7E    34 A5    14   C1   0103              ADDL3   #VCB$T_VOLNAME,UCB$L_VCB(R5),-(SP) ; save address of volume label
       56    14    28 A5   C1   0108              ADDL3   UCB$L_DDB(R5),#DDB$T_NAME,R6 ; calculate device name address
                50    66   9A   010D              MOVZBL  (R6),R0                         ; get length of device name
                      50   D6   0110              INCL    R0                              ; bump to include count byte
  62    10    00    66 50  2C   0112              MOVC5   R0,(R6),#0,#16,(R2)             ; copy device name into message
                      83   D4   0118              CLRL    (R3)+                           ; zero rel vol number and volume set size
          63    9E    0C   28   011A              MOVC3   #12,@(SP)+,(R3)                 ; copy volume label
                      04   BA   011E              POPR    #^M<R2>                         ; recover buffer address
                   FEDD'   30   0120              BSBW    ERL$RELEASEMB                   ; release error log buffer and send
                      38   BA   0123              POPR    #^M<R3,R4,R5>
                                0125   ;
                                0125   ; Release the device, using an unload and/or available function,
                                0125   ; depending on whether the volume is supposed to be unloaded or not.
                                0125   ;
                      52   D4   0125   10$:       CLRL    R2                              ; assume privilege bit clear
       02 0084 C4     16   E3   0127              BBCS    #PRV$V_PHY_IO,PCB$Q_PRIV(R4),20$ ; set PHY_IO privilege and test
                      52   D6   012D              INCL    R2                              ; bit was set - save state
          09 64 A5    0C   E5   012F   20$:       BBCC    #UCB$V_UNLOAD,UCB$W_STS(R5),30$ ; branch if no unload
                      01   DD   0134              PUSHL   #IO$_UNLOAD                     ; set up for unload
                      63   DD   0136              PUSHL   CHANNEL(R3)                     ; push channel number
             0255'CF  02   FB   0138              CALLS   #2,W^DO_IO                      ; issue the unload or rewind QIO
                      11   DD   013D   30$:       PUSHL   #IO$_AVAILABLE                  ; now release drive
                      63   DD   013F              PUSHL   CHANNEL(R3)                     ; push channel number
             0255'CF  02   FB   0141              CALLS   #2,W^DO_IO                      ; issue the unload or rewind QIO
  0084 C4    01   16  52   F0   0146              INSV    R2,#PRV$V_PHY_IO,#1,PCB$Q_PRIV(R4) ; restore privilege bit
```

```
      0800 8F      AA  014D          BICW    #UCB$M_VALID,-              ; clear software volume valid.
         64 A5         0151                  UCB$W_STS(R5)
                       0153      ;
                       0153      ; Now complete the dismount. If the device is cluster accessible, raise
                       0153      ; the device lock to read the value block.
                       0153      ;
      0088 8F      BB  0153          PUSHR   #^M<R3,R7>                  ; save R3 & R7
   57    34 A5     D0  0157          MOVL    UCB$L_VCB(R5),R7            ; get VCB address
   5E      18      C2  015B          SUBL    #24,SP                      ; allocate lock status block on stack
   56      5E      D0  015E          MOVL    SP,R6
04 A6    20 A5     D0  0161          MOVL    UCB$L_LOCKID(R5),4(R6)      ; get device lock ID
           6C      13  0166          BEQL    50$                         ; branch if none - not cluster dev
61 38 A5   17      E0  0168          BBS     #DEV$V_ALL,UCB$L_DEVCHAR(R5),40$ ; branch if dev allocated
   50      04      D0  016D          MOVL    #LCK$K_PWMODE,R0            ; otherwise raise lock to PW
                       0170          $ENQW_S LKMODE=R0,-                 ; queue for the device lock
                       0170                  LKSB=(R6),-
                       0170                  EFN=S^#EXE$C_SYSEFN,-
                       0170                  FLAGS=#LCK$M_CONVERT!LCK$M_VALBLK!LCK$M_SYNCSTS!LCK$M_NOQUOTA
         61 50     E9  0189          BLBC    R0,LOCKERR                  ; bug check if error
         09 66     E8  018C          BLBS    (R6),35$
   09F0 8F 66      B1  018F          CMPW    (R6), #SS$_VALNOTVALID      ; Is the error simply value block not valid?
           02      13  0194          BEQL    35$                         ; No problem.
           55      11  0196          BRB     LOCKERR                     ; Problem.
                       0198      ;
                       0198      ; Now get the lock count on the volume lock. If it is about to go to
                       0198      ; zero, clear the value block in the device lock.
                       0198      ;
           7E      D4  0198  35$:    CLRL    -(SP)                       ; longword for lock count
           7E      7C  019A          CLRQ    -(SP)                       ; item list end + retlen
        08 AE      9F  019C          PUSHAB  8(SP)                       ; address of block count
  02050004 8F      DD  019F          PUSHL   #LKI$_LCKCOUNT@16!4         ; size & item code for lock count
   51      5E      D0  01A5          MOVL    SP,R1                       ; item list address
           7E      7C  01A8          CLRQ    -(SP)                       ; IOSB
   50      5E      D0  01AA          MOVL    SP,R0
                       01AD          $GETLKIW_S      LKIDADR=VCB$L_VOLLKID(R7),-
                       01AD                          ITMLST=(R1),-
                       01AD                          EFN=S^#EXE$C_SYSEFN,-
                       01AD                          IOSB=(R0)
         29 50     E9  01C1          BLBC    R0,LOCKERR                  ; bug check if error
         26 6E     E9  01C4          BLBC    (SP),LOCKERR
   5E      18      C0  01C7          ADDL    #24,SP                      ; clean IOSB & item list off stack
           8E      D7  01CA          DECL    (SP)+                       ; check lock count against 1
           06      12  01CC          BNEQ    50$                         ; branch if other mounts exist
        08 A6      7C  01CE  40$:    CLRQ    8(R6)                       ; last mount - clear value block
        10 A6      7C  01D1          CLRQ    16(R6)
                       01D4      ;
                       01D4      ; Now take out the I/O database mutex again, and clean out the mount.
                       01D4      ; Release the volume lock if there is one.
                       01D4      ;
       FE29'       30  01D4  50$:    BSBW    SCH$IOLOCKW                 ; take I/O database mutex
   50    7C A7     D0  01D7          MOVL    VCB$L_VOLLKID(R7),R0        ; get volume lock ID
           14      13  01DB          BEQL    60$                         ; branch if none
                       01DD          $DEQ_S  LKID=R0                     ; release it
         04 50     E8  01EA          BLBS    R0,60$                      ; branch if OK
                       01ED      ;
                       01ED      ; To here on any errors from lock management services.
                       01ED      ;
```

```
                             01ED  LOCKERR:
                             01ED          BUG_CHECK XQPERR,FATAL          ; unexpected lock manager error
                             01F1  ;
                             01F1  ; Clear out the UCB.
                             01F1  ;
                        CA   01F1  60$:    BICL    #<DEV$M_DMT!DEV$M_FOR!- ; clear marked for dismount, foreign,
                             01F2                  DEV$M_RCK!DEV$M_WCK!-   ; read/write check,
                             01F2                  DEV$M_SWL!DEV$M_MNT>,-  ; software write locked, and mounted
 38 A5    C3280000 8F        01F2                  UCB$L_DEVCHAR(R5)       ; status bit.
             5C A5      B7   01F9          DECW    UCB$W_REFC(R5)          ; remove mount from ref count
          50    57      D0   01FC          MOVL    R7,R0                   ; get address of VCB.
             34 A5      D4   01FF          CLRL    UCB$L_VCB(R5)           ; clear address of VCB.
              FDFB'     30   0202          BSBW    EXE$DEANONPAGED         ; deallocate VCB.
      50     1C A5      D0   0205          MOVL    UCB$L_ORB(R5),R0        ; get the ORB address
                             0209
                             0209          ASSUME  ORB$L_OWN_PROT EQ ORB$L_SYS_PROT+4
                             0209          ASSUME  ORB$L_WOR_PROT EQ ORB$L_GRP_PROT+4
                             0209
             18 A0      7C   0209          CLRQ    ORB$L_SYS_PROT(R0)      ; clear out stale protection info
             20 A0      7C   020C          CLRQ    ORB$L_GRP_PROT(R0)
                60      D4   020F          CLRL    ORB$L_OWNER(R0)         ; clear out stale owner also
                             0211  ;
                             0211  ; Release the device lock with the updated value block.
                             0211  ;
             04 A6      D5   0211          TSTL    4(R6)                   ; check if we have a lock ID
                2E      13   0214          BEQL    80$                     ; branch if no lock
          50    01      D0   0216          MOVL    #LCK$K_CRMODE,R0        ; use CR mode if not allocated
    03 38 A5    17      E1   0219          BBC     #DEV$V_ALL,UCB$L_DEVCHAR(R5),70$ ; branch if not allocated
          50    05      D0   021E          MOVL    #LCK$K_EXMODE,R0        ; use EX mode if allocated
                             0221  70$:    $ENQW_S LKMODE=R0,-             ; convert the device lock down
                             0221                  LKSB=(R6),-             ; writing possibly modified value block
                             0221                  EFN=S^#EXE$C_SYSEFN,-
                             0221  FLAGS=#LCK$M_CONVERT!LCK$M_CVTSYS!LCK$M_VALBLK!LCK$M_SYNCSTS!LCK$M_NOQUOTA
                             023E  ; Sorry about the tacky format above, but the assembler won't parse
                             023E  ; macro args broken across lines.
          AC 50         E9   023E          BLBC    R0,LOCKERR              ; bug check if error
          A9 66         E9   0241          BLBC    (R6),LOCKERR
                             0244  ;
                             0244  ; Call routine to deallocate the device when appropriate
                             0244  ;
              FDB9'     30   0244  80$:    BSBW    IOC$DALLOC_DMT          ; complete the deallocation now
                             0247
              FDB6'     30   0247  90$:    BSBW    SCH$IOUNLOCK            ; release the I/O database mutex
                             024A          SETIPL  #0                      ; and drop IPL
          5E    18      C0   024D          ADDL    #24,SP                  ; clean the stack
        0088 8F         BA   0250          POPR    #^M<R3,R7>              ; restore R3 & R7
                        05   0254          RSB
                             0255
                             0255          .DISABLE LSB
```

```
                    0255          .SBTTL  DO_IO - COMMON I/O ROUTINE
                    0255  ;++
                    0255  ; DO_IO
                    0255  ;
                    0255  ; FUNCTIONAL DESCRIPTION:
                    0255  ;
                    0255  ;       This routine is an envelope procedure for all I/O done by this
                    0255  ;       module.  Use a system event flag for the I/O.  Since $QIOW now
                    0255  ;       properly waits for the combination of the event flag and IOSB
                    0255  ;       to be set, no special synchronization is needed here.
                    0255  ;
                    0255  ; INPUT:
                    0255  ;
                    0255  ;       CHAN(AP) = channel number to use for the I/O
                    0255  ;       FUNC(AP) = I/O function code
                    0255  ;
                    0255  ; OUTPUT:
                    0255  ;
                    0255  ;       NONE.
                    0255  ;
                    0255  ; ROUTINE VALUE:
                    0255  ;
                    0255  ;       R0      = some system status code
                    0255  ;--
                    0255
                    0255  ;
                    0255  ; Useful symbols
                    0255  ;
                    0255
          00000004  0255  CHAN = 4                                          ; offset to channel number
          00000008  0255  FUNC = 8                                          ; offset to I/O function code
                    0255
                    0255
              0004  0255  DO_IO:  .WORD   ^M<R2>                            ; common I/O routine
     52   7E    7E  0257          MOVAQ   -(SP),R2                          ; reserve IOSB, address to R2
                    025A          $QIOW_S EFN=S^#EXE$C_SYSEFN,-             ; use system event flag
                    025A                  CHAN=CHAN(AP),-                   ; use channel supplied by caller
                    025A                  FUNC=FUNC(AP),-                   ; use function code supplied by caller
                    025A                  IOSB=(R2)                         ; use local IOSB
        03 50   E9  0277          BLBC    R0,10$                            ; branch if error
     50   62    3C  027A          MOVZWL  (R2),R0                           ; set the return status in R0
                04  027D  10$:    RET                                       ; return
```

```
                    027E                .SBTTL  DELETE_RUJ - DELETE RECOVERY UNIT JOURNAL
                    027E         ;++
                    027E         ; DELETE_RUJ
                    027E         ;
                    027E         ; FUNCTIONAL DESCRIPTION:
                    027E         ;
                    027E         ;      Delete the recovery unit journal on this volume.
                    027E         ;      Failure to do so will leave the journal file open
                    027E         ;      and the device marked for dismount.  This routine
                    027E         ;      must be called after the dismount $QIO has been
                    027E         ;      sent to the ACP.
                    027E         ;
                    027E         ; INPUT:
                    027E         ;
                    027E         ;      R4 = address of device name descriptor
                    027E         ;      R5 = device UCB address
                    027E         ;
                    027E         ; OUTPUT:
                    027E         ;
                    027E         ;      NONE.   (Contents of R0 and R1 are unpredictable)
                    027E         ;
                    027E         ; ROUTINE VALUE:
                    027E         ;
                    027E         ;      NONE.
                    027E         ;--
                    027E
                    027E        DELETE_RUJ:                                             ; delete recovery unit journal
         05     E0  027E                BBS     #DEV$V_SQD,-                            ; only disks have RUJ's
   41 38 A5         0280                        UCB$L_DEVCHAR(R5),20$                   ;
                    0283         ;
                    0283         ; Assign a channel to the RUJ.  If the service fails,
                    0283         ; exit immediately, as it means that no RUJ is active.
                    0283         ;
         05     BB  0283                PUSHR   #^M<R0,R2>                              ; save R2 and make local storage
   52    5E     D0  0285                MOVL    SP,R2                                   ; save SP
                    0288                $ASSJNL_S CHAN   = (R2),-                       ; channel to journal
                    0288                          JNLTYP = #DT$_RUJNL,-                 ; journal type
                    0288                          DEVNAM = (R4)                         ; device name descriptor
      1B 50     E9  02A4                BLBC    R0,10$                                  ; branch if error
                    02A7         ;
                    02A7         ; Delete the journal.  The channel to the journal is
                    02A7         ; deassigned in the process.
                    02A7         ;
                    02A7                $DELJNL_S CHAN  = (R2)                          ; delete the journal
      0C 50     E8  02B3                BLBS    R0, 10$                                 ; if success then deljnl
                    02B6                                                                ;  deassigned the channel for us
                    02B6                $DEASJNL_S CHAN = (R2)                          ; deassign the journal channel
         05     BA  02C2        10$:    POPR    #^M<R0,R2>                              ; restore stack
                05  02C4        20$:    RSB                                             ; return
                    02C5
                    02C5
                    02C5                .END
```

```
                                                          E  4
DISMOUNT                    - DISMOUNT A MOUNTED MASS STORAGE VOLUME 16-SEP-1984 00:02:34  VAX/VMS Macro V04-00        Page  13
Symbol table                                                          5-SEP-1984 03:41:26  [SYS.SRC]DISMOUNT.MAR;1          (5)

$$T1                         = 00000000               LCK$M_SYNCSTS                = 00000008
BUG$_XQPERR                    ********   X    02     LCK$M_VALBLK                 = 00000001
CHAN                         = 00000004               LKI$_LCKCOUNT                = 00000205
CHANNEL                      = 00000000               LNM$DELETE_LNMB                ********   X    02
CJF$ASSJNL                     ********  GX    02     LNM$LOCKW                      ********   X    02
CJF$DEASJNL                    ********  GX    02     LNM$UNLOCK                     ********   X    02
CJF$DELJNL                     ********  GX    02     LOCAL_SIZE                   = 0000002C
CTL$GQ_ALLOCREG                ********   X    02     LOCKERR                        000001ED R      02
DDB$T_NAME                   = 00000014               MTL$B_STATUS                 = 0000000B
DELETE_RUJ                     0000027E R      02     MTL$L_LOGNAME                = 00000010
DEV$M_DMT                    = 00200000               MTL$L_UCB                    = 0000000C
DEV$M_FOR                    = 01000000               MTL$V_VOLSET                 = 00000000
DEV$M_MNT                    = 00080000               MTL$W_SIZE                   = 00000008
DEV$M_RCK                    = 40000000               NAME_LENGTH                  = 00000020
DEV$M_SWL                    = 02000000               NAME_STRING                  = 0000000C
DEV$M_WCK                    = 80000000               ORB$L_GRP_PROT               = 00000020
DEV$V_ALL                    = 00000017               ORB$L_OWNER                  = 00000000
DEV$V_DMT                    = 00000015               ORB$L_OWN_PROT               = 0000001C
DEV$V_FOR                    = 00000018               ORB$L_SYS_PROT               = 00000018
DEV$V_SQD                    = 00000005               ORB$L_WOR_PROT               = 00000024
DEVICE_NAME                  = 00000004               PCB$Q_PRIV                   = 00000084
DO_IO                          00000255 R      02     PR$_IPL                      = 00000012
DT$_RUJNL                    = 00000001               PRV$V_PHY_IO                 = 00000016
EMB$B_VM_NAMLNG              = 0000001E               SCH$IOLOCKW                    ********   X    02
EMB$K_VD                     = 00000041               SCH$IOUNLOCK                   ********   X    02
EMB$K_VM_LENGTH              = 0000003E               SS$_NORMAL                   = 00000001
EMB$L_VM_ERRCNT              = 00000014               SS$_VALNOTVALID              = 000009F0
EMB$L_VM_OPRCNT              = 00000018               SYS$ASSIGN                     ********  GX    02
EMB$L_VM_OWNUIC              = 00000010               SYS$DASSGN                     ********  GX    02
EMB$T_VM_LABEL               = 00000032               SYS$DEQ                        ********  GX    02
EMB$T_VM_NAMTXT              = 0000001F               SYS$ENQW                       ********  GX    02
EMB$W_VM_ENTRY               = 00000004               SYS$GETLKIW                    ********  GX    02
EMB$W_VM_NUMSET              = 00000030               SYS$QIOW                       ********  GX    02
EMB$W_VM_UNIT                = 0000001C               UCB$L_DDB                    = 00000028
EMB$W_VM_VOLNUM              = 0000002E               UCB$L_DEVCHAR                = 00000038
ERL$ALLOCEMB                   ********   X    02     UCB$L_LOCKID                 = 00000020
ERL$RELEASEMB                  ********   X    02     UCB$L_OPCNT                  = 00000070
EXE$C_SYSEFN                   ********   X    02     UCB$L_ORB                    = 0000001C
EXE$DEALLOCATE                 ********   X    02     UCB$L_VCB                    = 00000034
EXE$DEANONPAGED                ********   X    02     UCB$M_VALID                  = 00000800
EXE$DEAPAGED                   ********   X    02     UCB$V_MOUNTING               = 00000009
FOREIGN                        000000DB R      02     UCB$V_UNLOAD                 = 0000000C
FUNC                         = 00000008               UCB$W_ERRCNT                 = 00000082
IO$M_DMOUNT                  = 00000400               UCB$W_REFC                   = 0000005C
IO$_ACPCONTROL               = 00000038               UCB$W_STS                    = 00000064
IO$_AVAILABLE                = 00000011               UCB$W_UNIT                   = 00000054
IO$_UNLOAD                   = 00000001               VCB$B_STATUS2                = 00000053
IOC$CVT_DEVNAM                 ********   X    02     VCB$L_VOLLKID                = 0000007C
IOC$DALLOC_DMT                 ********   X    02     VCB$T_VOLNAME                = 00000014
IOC$DISMOUNT                   00000000 RG     02     VCB$V_MOUNTVER               = 00000002
IPL$_ASTDEL                  = 00000002               VCB$W_MCOUNT                 = 0000004C
LCK$K_CRMODE                 = 00000001
LCK$K_EXMODE                 = 00000005
LCK$K_PWMODE                 = 00000004
LCK$M_CONVERT                = 00000002
LCK$M_CVTSYS                 = 00000040
LCK$M_NOQUOTA                = 00000020
```

```
                                                 F 4
DISMOUNT                        - DISMOUNT A MOUNTED MASS STORAGE VOLUME 16-SEP-1984 00:02:34  VAX/VMS Macro V04-00          Page  14
Psect synopsis                                                            5-SEP-1984 03:41:26  [SYS.SRC]DISMOUNT.MAR;1              (5)

                                         +----------------+
                                         ! Psect synopsis !
                                         +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00000000  (    0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
Y$DISMOUNT                      000002C5  (  709.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                         +------------------------+
                                         ! Performance indicators !
                                         +------------------------+

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        29    00:00:00.09    00:00:01.35
Command processing                   115    00:00:00.54    00:00:04.35
Pass 1                               493    00:00:19.98    00:01:03.98
Symbol table sort                      1    00:00:03.25    00:00:10.29
Pass 2                               115    00:00:03.44    00:00:10.17
Symbol table output                   13    00:00:00.12    00:00:00.53
Psect synopsis output                  2    00:00:00.02    00:00:00.02
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 770    00:00:27.44    00:01:30.69

The working set limit was 1650 pages.
113983 bytes (223 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2155 non-local and 26 local symbols.
560 source lines were read in Pass 1, producing 15 object records in Pass 2.
42 pages of virtual memory were used to define 41 macros.

                                         +--------------------------+
                                         ! Macro library statistics !
                                         +--------------------------+

Macro library name                          Macros defined
------------------                          --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                         13
_$255$DUA28:[SYSLIB]STARLET.MLB;2                      25
TOTALS (all libraries)                                 38

2412 GETS were required to define 38 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:DISMOUNT/OBJ=OBJ$:DISMOUNT MSRC$:DISMOUNT/UPDATE=(ENH$:DISMOUNT)+EXECML$/LIB
```

0374 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY